Industrial Ethernet Security Harmonization Group

_____

WHITEPAPER ON HUMAN USER AUTHENTICATION IN THE OT ENVIRONMENT

Version 1.0 | August 12th, 2025

Disclaimer

Prepared by Industrial Ethernet Security Harmonization Group, consisting of the Standards Developing Organizations (SDOs):

- FCG (FieldComm Group
- ODVA, Inc.
- OPC Foundation
- PI (PROFIBUS&PROFINET International))

Core group members contributing (alphabetic order):

Andreas Walz (PI)
Dominik Ziegler (PI)
Frank Fengler (FCG)
Jack Visoky (ODVA)
Joakim Wiberg (ODVA)
Randy Armstrong (OPC Foundation)
Simon Merklin (PI)
Stephen Mitschke (FCG)

Comments to be submitted to working group editor: simon.merklin@endress.com

Special thanks to the NAMUR Working Group AK 4.18.4, supporting the creation of this document.

Introduction

Authentication (who is it?) and authorization (what are they allowed to do?) of users is a fundamental aspect of cybersecurity. However, there have been challenges associated with deploying and using authentication and authorization in the Operational Technology (OT) environment. Despite this, many organizations and vendors are working to provide technologies that bring user authentication to the OT environment. This whitepaper explores some of the challenges around authentication and authorization in the OT environment, technologies that can be used, and key use cases. It provides an overview of the landscape and possibilities but is not meant to specify details of design or implementation.

Intended Audience

The intended audience for this whitepaper includes users of OT products with some background in security as well as OT product vendors. It is not intended to target security experts, as the explanations are high level and lacking in specific details. However, the intended audience should at least have a rudimentary understanding of basic cybersecurity concepts. This whitepaper is not meant as an academic paper, rather it is meant to provide practical advice for users and vendors.

Scope

The scope of this paper is to describe, from a high-level, Human User Authentication and Authorization within the OT environment. Workflows are described along with technologies that enable that workflow. However, there will necessarily be aspects of authentication that are specific to the various OT protocols, equipment, and installations that cannot be covered by a whitepaper such as this. The provisioning and bootstrapping of the equipment and software is also outside of the scope of this whitepaper.

Challenges and constraints

Deploying and using authentication in the OT environment does present some challenges beyond what might be found in a typical Information technology (IT) use case. A few of these challenges are:

- Equipment as well as users may be connected or disconnected from the network that provides authentication
- Time synchronization may not be available or supported
- Administrators often want to manage users centrally
- Authentication and authorization should work independently of the interface used (e.g. wired, wireless, Bluetooth, etc.)
- It is not desirable to transmit confidential credentials like passwords to every OT device
- For highly constrained (non-ethernet) field devices, today no public key infrastructure (PKI) solution exists.
- For connected environments, integration with existing IT tools is desirable
- Multifactor authentication is desired, but it is challenging for OT devices to support this directly
- Highly heterogeneous environment with products from various vendors that do not necessarily interoperate

- Some devices operate on highly constrained platforms with little memory or other computing resources (e.g. temperature sensors with HART and Bluetooth Low Energy), but still need to support authentication/authorization
- Emergency access to equipment can be necessary even when a central Identity Provider is not available
- On and offsite repairs, initial commissioning
- As a secondary concern, in many situations traceability of events like authentication is important
- Many devices do not have a user interface

Technology Solutions

Fortunately, some existing standards can provide a basis for meeting these challenges. Modern IT environments generally provide a mechanism for issuing digitally signed tokens to an application after a user is authenticated. These tokens provide proof of authentication that can be presented to devices that the user needs to access. These tokens contain information that can be used to determine what permissions, if any, are granted to the application providing the token. This information includes information such as the user identity, group membership, target resource, etc.

Table 1 General Comparison of Tokens and Passwords

General Comparison of Tokens and Passwords		
	Tokens	**Passwords**
Lifetime	Short, usually minutes, hours, or days	Long, usually months or years
Generation	Machine generated	Human generated (exception for password managers)
Credential Management	Software managed, transparent to the user	User managed, need to be remembered or stored in a user accessible manner
Extensibility	Claims can be embedded that represent any data necessary	Not extensible (single piece of data)
Usage with Authorization	Can communicate authorization information like role assignments.	Cannot communicate additional authorization information.
Account Management	Can be managed by a central Identity Provider that is transparent to any device; trust is just needed for the Identity Provider	Typically, needs to be communicated directly to the device that is authenticating the user

Although there are a variety of ways that this can be done, one of the most common is using a technology called "OpenID Connect". OpenID Connect uses JSON Web Tokens (JWTs) as proof of authentication and is built upon OAuth 2.1 authorization flows. The available authorization flows are described in Table 2.

Table 2 - OAuth 2.1 authorization flows

Name	**Use Cases**
Authorization Code Flow with PKCE	Browser based web applications and mobile applications. Operators that use mobile devices to manage equipment will use this flow with their personal credentials.
Client Credentials Flow	Device to device communication without a user. Centralized network management applications or purpose-built mobile devices will use this flow with pre-installed secrets.
Device Authorization Flow	Devices with minimal displays that can display a code. Not a good fit for the OT use cases identified in this document.

A concrete example of the authorization code flow is used with many websites where a user can sign-in to a site using credentials from another Identity Management system. For example:

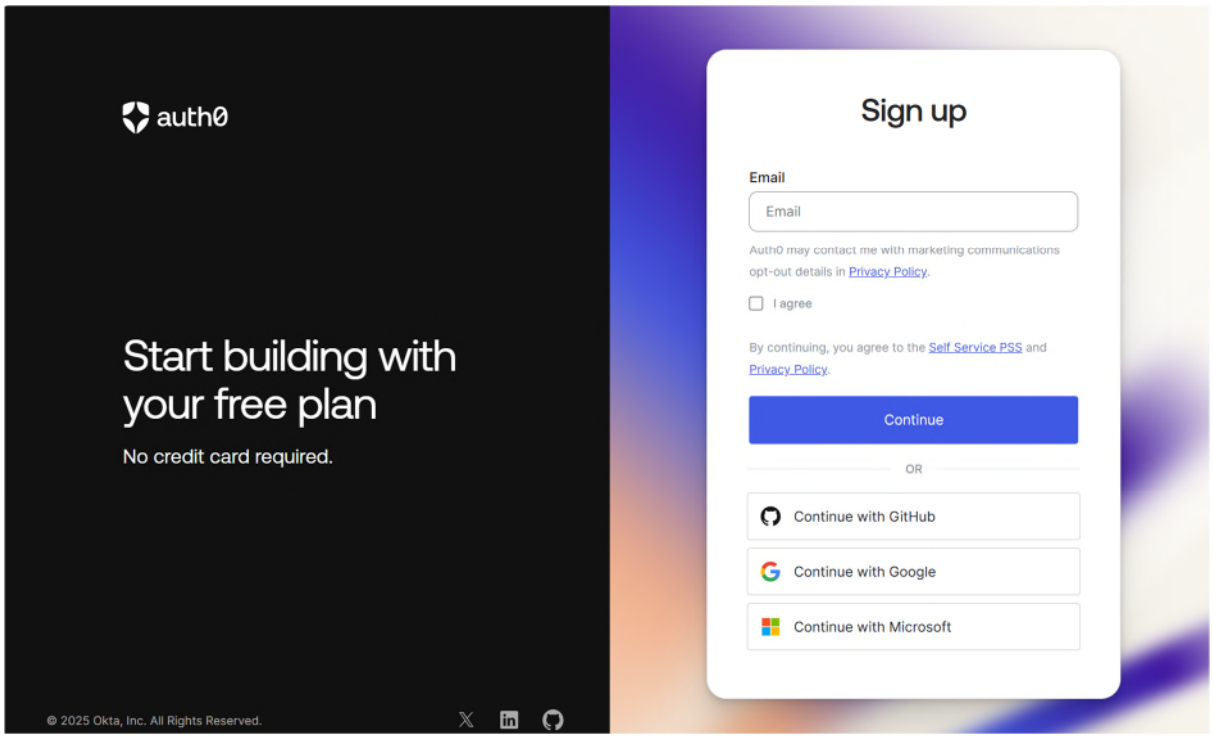

Figure 1: Example website implementing OpenID Connect Authentication workflows.

The site “auth0.com” allows for sign in directly, or via GitHub, Google, or Microsoft’s identity management system, which provides an Access Token as proof of that authentication (in this case via an OpenID Connect workflow). There are a few basic points to keep in mind which are common to token-based authentication schemes:

1. There is a central identity management system where users are managed
2. Authentication occurs via this system and can support many different workflows (e.g., multifactor authentication)
3. No matter what authentication mechanism is used, an “Access Token” is produced. This token is typically a JSON Web Token (JWT) and contains information about the user and can be used as proof of authentication when accessing a device or server.
4. The Access Token also restricts where and how it can be used by:
    a. Specifying the target Device(s) via the `‘aud’ (audience)` claim
    b. Specifying the permissions that are needed via Roles via any custom role claims
    c. Specifying a validity time via the `‘nbf’(not before)` and `‘exp’` (expiration time) claim

Preconditions:

1. An Identity Management System is set up (e.g. OpenID Connect)

2. OT devices, equipment, and software are configured to have a trust relationship to the Identity Management System; specifically, the public key for verifying token signatures is distributed to these OT assets.
3. Users authenticate with the Identity Management System (could require multifactor authentication).
4. Users then receive one or more Access Tokens; these tokens communicate identity and role information when accessing OT equipment, devices and software. In other words,, the OT API (e.g. EtherNet/IP, HART-IP, OPC UA, PROFINET, etc.) trusts and consumes the Access Token as proof of authentication and provides appropriate privilege levels per the given token.

Constraints

To meet the OT environment's specific authentication challenges, OpenID Connect would have to have to address these issues:

- Equipment/users may be connected or disconnected
    - Access Tokens are cached in the authentication client and can be sent even in a disconnected environment.
- Time synchronization may not be available or supported
    - If available time synchronization can be used directly. If not, devices can make use of a Real Time Clock (RTC) that has been set at device provisioning and can get updates whenever an authenticated client connects to them (e.g. mobile phone).
      If time synchronization is, in the worst case, not available, the system should still be usable, however, some limitations apply. A recovery mechanism may be needed for the security configuration. It is important if the time can no longer be synchronized.
- Administrators often want to manage users centrally
    - Authentication servers that produce tokens allow users to be managed directly on the server, without needing to update every device when the database of users is modified.
- Authentication/authorization should work independent of the interface used (e.g. wired, wireless, Bluetooth, etc.)
    - Tokens can be transmitted on any digital communication interface
- It is not desirable to transmit credentials like passwords to every OT device
    - Tokens abstract away credentials and allow for a "proof" of authentication to be presented. Tokens can be limited to a certain "audience" and even if stolen are time limited (expire)
- For connected environments, integration with existing IT tools is desirable
    - Open ID Connect and other token-based technologies are widely deployed in IT environments, with many commercial and open-source products available
- Multifactor authentication is desired, however, it is challenging for devices to support this directly
    - Using a centralized server for authentication allows for a wide variety of authentication workflows, without the need for the device to know how this works
- Highly heterogeneous environment with products from various vendors that do not necessarily interoperate

    - Tokens offer an abstraction of protocols and vendor-specific technology; this allows a single Identity Provider to serve tokens for multiple OT protocols or devices from various OT vendors
- Some devices operate on highly constrained platforms with little memory or other computing resources (e.g. temperature sensors with Ethernet-APL), but still need to support authentication/authorization
    - Processing of a token is relatively lightweight and reduces the need of the device to understand the details of authentication; essentially, token processing will require parsing and signature verification, which can be achieved in constrained devices.
    - There might be additional processing, like the validation of the certificate trust chain (PKI) and additional bearer authentication if a mitigation to token theft is needed.
- Emergency access to equipment can be necessary when a central Identity Provider is not available
    - An Access Token with emergency access rights can be stored in a well-known location that is always accessible even if the external network is down. Access to these tokens could be protected with a password that is stored in a physical 'break glass in case of emergency' container. There are different strategies that can be used to secure these Access Tokens. For example, Access Tokens may be signed with a special key that is only used for these types of tokens and is revoked after an emergency. They may have a longer expiry period when combined with a business process to ensure valid tokens are always available.
- As a secondary concern, the traceability of events like authentication is important
    - Logging is already implemented in most Identity Providers, and can optionally be done on the OT device itself
- Many devices do not have a local user interface
    - The token-based scheme allows for the user interface to be implemented by the identity provider and any software that interacts with it, leaving the device free to continue to operate without a local user interface

Use Cases

An example of a use case in the OT environment where this type of user authentication scheme might be applied is shown on the following figure. This figure shows the OT environment is not networked to the IT Identity Management System, but the user is able to authenticate in the IT environment and then use the token(s) as proof of their authentication in the OT environment. Of course, many variations of this are possible, including various levels of connectivity from the IT environment to the OT environment.

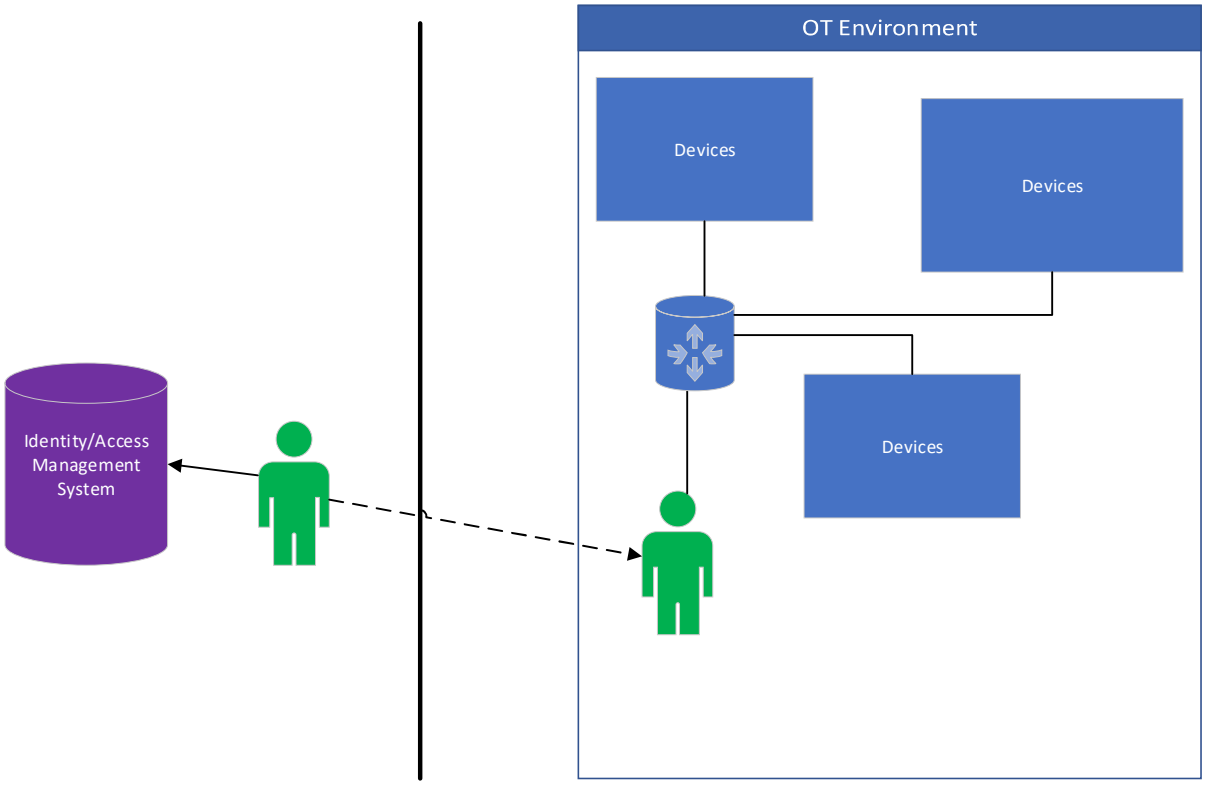

Figure 2: Use case for an OT Device on a Isolated Network

Figure 3: UML Sequence Diagram

shows the use case from NE201 Identitiy and Access Management on Automation Devices for an OT Device on an Isolated Network, disconnected from the Plant Network. The token could be distributed from a vendor app via BLE to low constraint OT Device, which gets on first use the plant trust anchor, which is used from now on for validating the token signature. Figure 3 provides an UML sequence diagram for this use case.

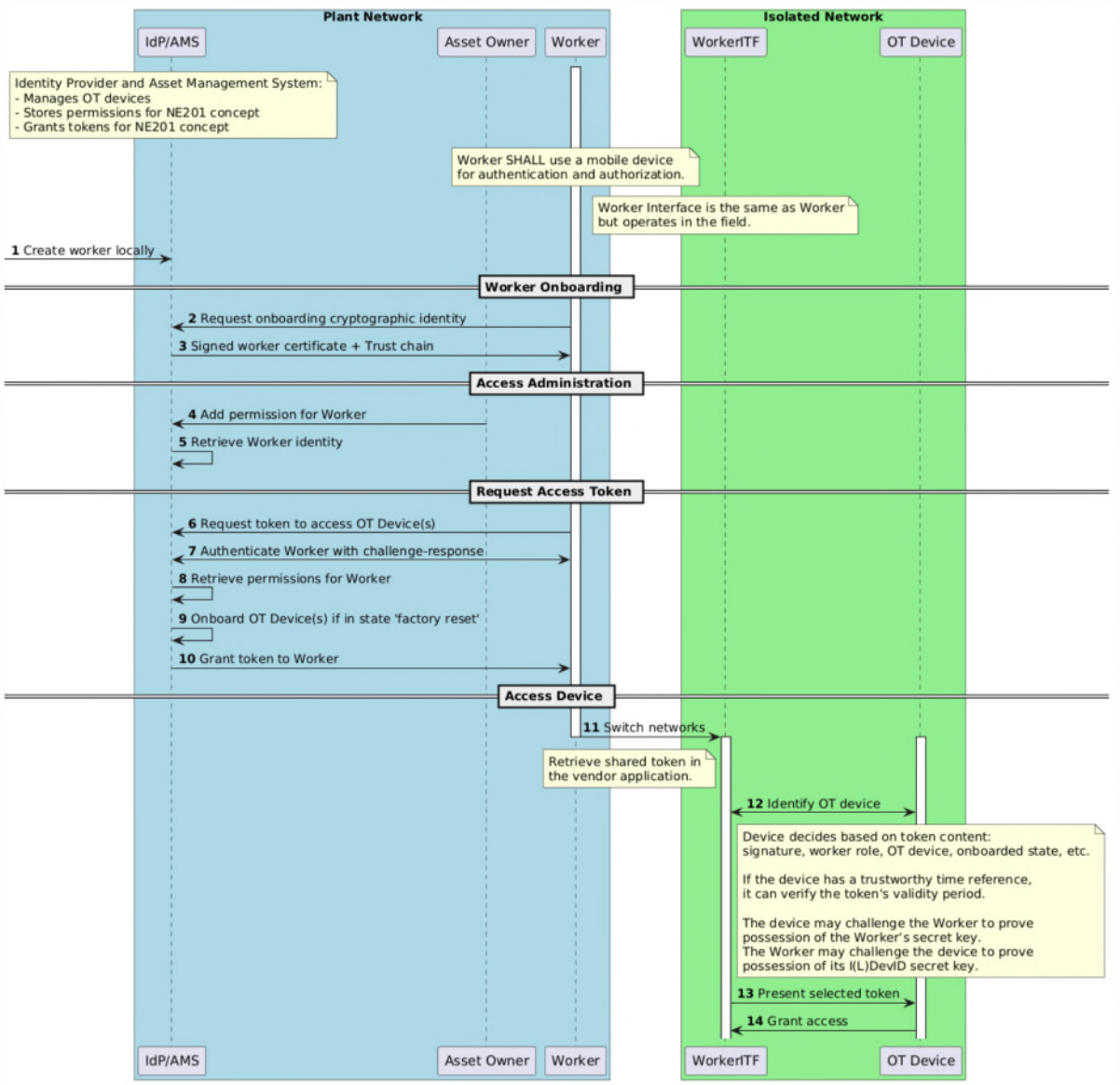

Figure 3: UML Sequence Diagram

Worker Onboarding

1. **Identify the Use Case**
    a. **Title**: Worker Onboarding
    b. **Objective**: The Worker obtains two factors for authentication and authorization. One is a set of customer-provided credentials and the second is a LDevID certificate granted to their mobile by the customer's infrastructure Identity Provider and Asset Management System(IdP_AMS) software.
2. **Define the Actors**
    a. **Primary Actor**: Designated Worker with own mobile device
    b. **Secondary Actors**: IIdP_AMS Software
3. **Describe the Preconditions**
    a. Designated_Worker has appropriate software on his mobile to execute this process
    b. IdP_AMS created the Designated_Worker locally with credentials

c. The mobile can create a private/public key pair, emit a Certificate Signing Request (CSR) and store a LDevID locally
d. The Designated_Worker (mobile) has an active communication link to the IdP_AMS

4. **Outline the Main Flow**
    a. **Steps**:
        i. Designated_Worker (mobile) requests onboarding to IdP_AMS and offers a CSR.
        ii. IdP_AMS registers the mobile and provides a signed Worker LDevID certificate and trust chain
        iii. Designated_Worker (mobile) stores the LDevID locally
    b. **Interactions**: Designated_Worker uses their own mobile device, enters their own credentials, which could be emailed securely beforehand and “clicks” to onboard their mobile.
5. **Specify the Postconditions**
    a. IdP_AMS has registered the mobile and granted it a signed LDevID
    b. A success message is shown to the Designated_Worker
6. **Include Alternative Flows**
    a. If preconditions cannot be fulfilled, or an abnormal condition occurs during the process then an error message with clear instructions is shown.

Access Administration

1. **Identify the Use Case**
    a. **Title**: Access Administration
    b. **Objective**: An Asset Owner creates a JSON Web Token (JWT) which contains all the required administrative and cryptographic data that an isolated device needs to accept and process the token. The IdP_AMS keeps track of the token.
2. **Define the Actors**
    a. **Primary Actor**: AssetOwner
    b. **Secondary Actors**: IdP_AMS Software
3. **Describe the Preconditions**
    a. Asset Owner is known to the IdP_AMS and has access to its software
    b. The target entitlements are known to the IdP_AMS
    c. The Designated_Worker (mobile) has been onboarded in the IdP_AMS
    d. EITHER the target OT_Device is known to the IdP_AMS e.g., Identification Link (IL) string (IEC 61406) within LDevID/IDevID
    OR the target OT_Device will be onboarded on a “Trust-On-First-Use" basis.
4. **Outline the Main Flow**
    a. **Steps**:
        i. Asset Owner requests entitlements for the Designated_Worker with all the required administrative information
        ii. The IdP_AMS records the request and registers the OT_Device if “Trust-on-First-Use" is activated
    b. **Interactions**:
        i. The Asset Owner enters credentials into the IdP_AMS, along with their request for entitlements.

ii. The IdP_AMS system provides guidance and records the appropriate information in its databases.

5. **Specify the Postconditions**
    a. The IdP_AMS has a record of entitlements available to a specific worker, for a specific OT_Device. Entitlements can be limited by various conditions (e.g.; IL string /Device serial number, worker opc-role defining allowed action(s), validity duration time, etc...).
6. **Include Alternative Flows**
    a. If preconditions cannot be fulfilled, or an abnormal condition occurs during the process then an error message with clear instructions is shown.

Device Onboarding (Trust-On-First-Use)

1. **Identify the Use Case**
    a. **Title**: Device Onboarding
    b. **Objective**: Ensure the IdP_AMS system only issues JWT tokens for approved OT_Devices. Provide a trust anchor to the OT_Device on subsequent first use of a JWT.
2. **Define the Actors**
    a. **Primary Actor**: OT_Device
    b. **Secondary Actors**: IdP_AMS
3. **Describe the Preconditions**
    a. The OT_Device is not already on-boarded in the IdP_AMS.
    b. IL string of the new OT_Device is known
    c. Asset Owner has performed an access administration procedure.
4. **Outline the Main Flow**
    a. **Steps**:
        i. A new IL string is provided to the IdP_AMS during an access administration procedure
        ii. IdP_AMS registers the new OT_Device in its database
        iii. IdP_AMS provides a trust anchor in the JWT token
        iv. On first use, the OT_Device will accept the JWT trust anchor as valid and store it for verification of later JWT tokens.
    b. **Interactions**: (Same as steps)
5. **Specify the Postconditions**
    a. The IdP_AMS will consider the OT_Device as valid in later requests for a JWT token.
    b. After First Use, the OT_Device stored the trust anchor and can use it to verify later JWT tokens.
6. **Include Alternative Flows**
    a. If preconditions cannot be fulfilled, or an abnormal condition occurs during the process then an error message with clear instructions is shown.

Device Onboarding (LDevID)

7. **Identify the Use Case**
    a. **Title**: Device Onboarding
    b. **Objective**: Ensure the IdP_AMS system only issues JWT tokens for approved OT_Devices. Provide a trust anchor to the OT_Device. Allow the Designated_Worker

(mobile) to challenge the OT_Device for proof-of-possession of the LDevID private key.

8. **Define the Actors**
    a. **Primary Actor**: OT_Device
    b. **Secondary Actors**: IdP_AMS, possibly via one or more mediators.
9. **Describe the Preconditions**
    a. The OT_Device is not already on-boarded in the IdP_AMS.
10. **Outline the Main Flow**
    a. **Steps**:
        i. OT_Device generates a Certificate Signing Request
        ii. CSR is transferred to the IdP_AMS via mediators
        iii. IdP_AMS signs the CSR and returns a LDevID certificate to the OT_Device
    b. **Interactions**: Describe the interactions between the actors and the system.
11. **Specify the Postconditions**
    a. IdP_AMS has LDevID of OT_Device and can include it in signed JWT tokens
    b. Worder_Designated (mobile) can use the LDevID in the JWT token to challenge the device
12. **Include Alternative Flows**
    a. If preconditions cannot be fulfilled, or an abnormal condition occurs during the process then an error message with clear instructions is shown.

Request Access Token

1. **Identify the Use Case**
    a. **Title**: Request Access Token
    b. **Objective**: An authorized person (Asset Owner or other) requests a JWT token valid for a specific Designated_Worker and with specific entitlements. A Designated_Worker could request an access token for himself, if authorized by the IdP_AMS.
2. **Define the Actors**
    a. **Primary Actor**: Authorized person / Designated_Worker
    b. **Secondary Actors**: IdP_AMS Software.
3. **Describe the Preconditions**
    a. All actors are already known by the IdP_AMS, in particular, the Designated_Worker (mobile) has been onboarded.
    b. All administrative information is already known by the IdP_AMS (entitlements, etc...)
    c. All relevant cryptographic information is already known by the IdP_AMS.
4. **Outline the Main Flow**
    a. **Steps**:
        i. Authorized person / Designated_Worker uses customer software to interact with their infrastructure
        ii. Authorized person / Designated_Worker inputs proper administrative information (credentials, etc...) and the system creates and stores a JWT token.
        iii. The system provides means for the authorized person to transfer the token to the Designated_Worker (mobile).
    b. **Interactions**:

        i. Same as <steps>.
5. **Specify the Postconditions**
    a. IdP_AMS has produced a JWT that contains all the administrative and cryptographic information an isolated device needs to assess its authenticity and act according to its content.
    b. IdP_AMS has a copy of the JWT in its database, ready to send by any appropriate means to requesters.
6. **Include Alternative Flows**
    a. If preconditions cannot be fulfilled, or an abnormal condition occurs during the process then an error message with clear instructions is shown.

Access Device

1. **Identify the Use Case**
    a. **Title**: Access Device
    b. **Objective**: A Designated_Worker (mobile) offers a JWT token to an isolated OT_Device. The OT_Device verifies the JWT as authentic and valid and executes administrative actions accordingly.
2. **Define the Actors**
    a. **Primary Actor**: OT_Device.
    b. **Secondary Actors**: Designated_Worker (mobile).
3. **Describe the Preconditions**
    a. The Designated_Worker (mobile) has already been provided with an authorized and valid JWT.
    b. The Designated_Worker (mobile) provides the JWT to the OT_Device via a mechanism for bi-directional communication with the OT_Device
    c. The OT_Device has a mechanism for assessing both authenticity and validity of the JWT.
4. **Outline the Main Flow**
    a. **Steps**:
        i. Designated_Worker (mobile) transfers JWT to OT_Device
        ii. OT_Device verifies the JWT (e.g.; IL string /Device serial number, worker opc-role defining allowed action(s), validity duration time, etc...)
        iii. OT_Device optionally challenges the Designated_Worker (mobile)
        iv. OT_Device is optionally challenged by the Designated_Worker (mobile)
        v. OT_Device executes actions according to the JWT administrative part.
    b. **Interactions**:
        i. Designated_Worker uses software on their mobile to send the JWT to the OT_Device
        ii. The OT_Device verifies the JWT and executes its administrative context.
5. **Specify the Postconditions**
    a. The OT_Device has executed the JWT administrative content
    b. The Designated_Worker (mobile) records the result of the interaction.
6. **Include Alternative Flows**
    a. If preconditions cannot be fulfilled, or an abnormal condition occurs during the process, then an error message with clear instructions is shown.

Access Tokens

Access Tokens have several features that are designed to enhance security by limiting the context where the tokens can be used. An example of an Access Token is given here. Note that a given Identity Provider might include more or less of these claims.

Example Access Token

```
Header:
    Message authentication code algorithm (alg) - m
    Key ID (kid) - m
    Token type (typ) - o -> JWT

Payload:
    Issued at (iat) -m
    valid to (exp) -m
    audience (aud) -m -> https://id.abb/9AAC129110?SN=3K650000554982
    subject (sub) -m -> identity with worker id
    opc_role -m
    token type (typ) -o -> Bearer
```

The features of an Access Token including optional ones are discussed in detail below.

Audience (aud) – RFC 7519

The Audience is the Device that is the intended consumer of the Access Token. It is identified by a system unique identifier such as the ProductInstanceUri (a.k.a. IL String) for the Device. If configured, then no Device should accept an Access Token that does not specify an Audience that refers to the Device.

Expiry Time (exp) – RFC 7519

The Expiry Time indicates when the Access Token expires and cannot be used. Devices with access to a synchronized clock should not accept an Access Token that has expired. Devices that do not have a synchronized clock can be configured to ignore this value and depend on other fields to protect against malicious actors.

Sometimes, an OT environment will have ‘emergency access tokens’ that are identified with roles or scopes. These tokens are stored in a secure location and are only used in an emergency.

x.509 Certificate Chain (x5c) – RFC 7517

A certificate chain in RFC4945 format corresponding to the private key used to generate the token signature. The Device uses this information to verify that the signature is valid and the token is authentic.

Client Certificate (cnf)- RFC 7800

A Device may issue a Proof-of-Possession challenge to the Presenter of a token if said token contains the required cryptographic material stored in its "cnf" claim (e.g. Ref. RFC-7800, "Figure 2" - Proof of Possession with an Asymmetric Key).

For example, the "cnf" claim could be a JSON Web key, either "EC" or "RSA". The “cnf” claim could also be an X509 certificate.

The “cnf” claim should only contain one type of cryptographic material (e.g. EITHER a key OR a certificate).

Role (opc_role) - Well Known Roles

Roles are used to enforce permissions within a Device. This allows users to be granted permissions by being assigned to a role. An Access Token will specify one or more system roles for that user. Users should be assigned the minimum roles needed to perform the task that they need to complete.

Devices are expected to have mappings between system roles and specific permissions within the device. Users will only be granted permissions available to Roles specified in the Access Tokens. In some cases, Devices will have internal Roles and maintain a mapping between system roles and internal roles.

A role represents a pre-defined set of parameters and/or functions that can be accessed by a human or machine user. The permissions of a role are assigned according to the tasks the role must perform in the OT equipment.

For further information, please refer to: Role-Based Access Control (auth0.com)

Roles are defined within a context. For example, most devices have the concept of an 'admin' role which grants administrator rights to users that belong to this role. The important feature of device specific roles is they are used to authorize actions on a device. System wide roles, on the other hand, are used to assign rights to users and are not associated with a device. Each device needs a way to either assign permissions to system wide roles or map them onto its device roles. Permissions are assigned to roles which are scoped to the audience for which the Access Token is intended.

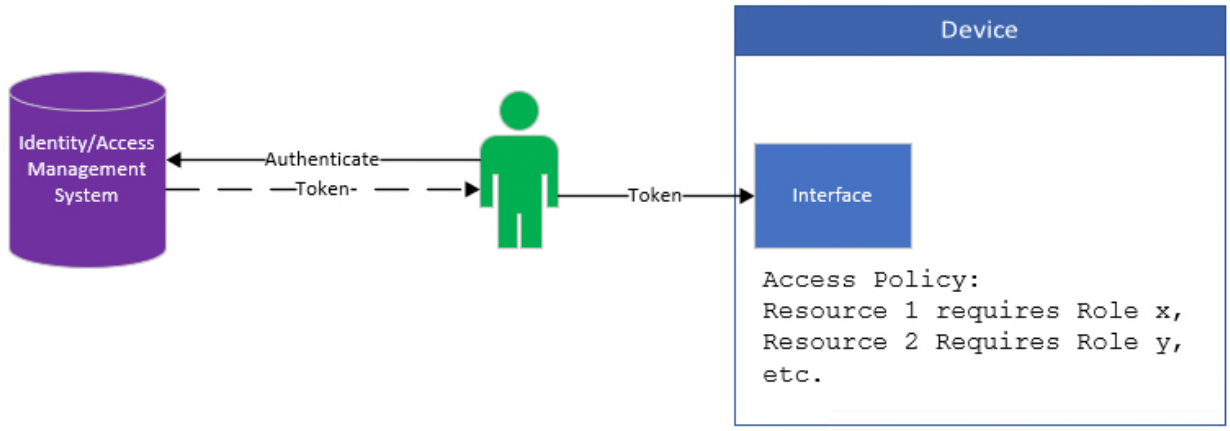

Figure 2: Access Policy applied within the protected resource.

Note that roles are not necessarily hierarchical, as some permissions for a role may overlap with other roles. The details for a given device/software/system are important and may vary.

In the following section, the "well-known role" concept is discussed. This concept can serve as the basis for implementations for Standards Development Organizations (SDOs) and vendor specific implementations.

OT-devices should provide a way to assign permissions to system wide roles by either mapping the system wide roles to internal Roles or via explicit permissions. The following well-known Roles are examples of possible system wide Roles that may need to be mapped to Device permissions.

Note that some Roles are ‘fallback’ Roles that are never specifically assigned to users or applications. They define the permissions available when a user and/or application meets certain criteria.

Role Examples

The following provides some examples of roles that may be found in an industrial system. Actual implementations may vary; it is important to understand specific distinctions of roles for a given system.

A freely available example of roles can be found in the OPC UA specification, UA Part 3: Address Space Model - 4.9.2 Well Known Roles and standard approach is shown in UA Part 18: Role-Based Security - 4 Role Model.

Role name: Observer

An Observer is only able to view the data and events produced by a Device.

The Role is allowed to browse, read data, read historical data/events or subscribe to data/events.

This Role cannot alter the state of the Device.

Role name: Operator

An Operator is generally only able to view the data and events produced by a Device, however, they may have access to limited ability to change the system such as the ability to acknowledge alarms.

The Operator is responsible for verifying that the system is running smoothly.

The Role is allowed to browse, read data, read historical data/events or subscribe to data/events.

In addition, the Session is allowed to write some live data and call some Methods.

Role name: Supervisor

This Role is allowed to browse, read live data, read historical data/events, call Methods or subscribe to data/events.

Role name: Engineer

An Engineer is able to view the data and events produced by a Device and make changes to operation parameters such as set points or the ability to change the PLC program.

The Engineer is responsible for setting up the system and making changes when required.

Role name: SecurityAdmin

The SecurityAdmin is responsible for all network and security related settings such as installing the keys needed to validate Access Tokens.

Role name: Anonymous

Anonymous is a fallback Role assigned when no recognized credentials are provided. It should not have any permissions that allow the state of a Device to be changed or allow access to sensitive information.

Role name: AuthenticatedUser

AuthenticatedUser is a fallback Role assigned when valid user credentials have been provided but no specific permissions have been granted.

Role name: AuthenticatedApplication

AuthenticatedApplication fallback Role assigned when the client application has been authenticated, however, no user credentials have been provided, or the user credentials have no specific permissions assigned.

Open questions to discuss as a community:

1. How can an administration of the services, products and users look like?
2. How can provisioning workflows look like?
3. How can we guarantee a shared and synchronized time source in the plant, independent of the interface?

Miscellaneous

Abbreviations

Abbreviation	Description
IESHG	Industrial Ethernet Security Harmonization Group
IT	Information technology
OT	Operation technology
SDO	Standards developing organization
PKI	Public key infrastructure
RTC	Real Time Clock

References

Introduction to Identity and Access Management (IAM) (auth0.com)

https://openid.net/developers/how-connect-works/

https://datatracker.ietf.org/doc/html/rfc6749

Version History

Version	Date	Changes
Version 1	12.08.2025	Release of first version

FieldComm Group

http://go.fieldcommgroup.org

ODVA

www.odva.org

OPC Foundation

www.opcfoundation.org

Profibus and Profinet International (PI)

www.profibus.com